University of South Bohemia in České Budějovice

Faculty of Science

and

Johannes Kepler University in Linz

Faculty of Engineering and Natural Sciences

Testing of different quality filtering algorithms on contig quality in plant transcriptome

Bachelor Thesis

Laura Kroll

Supervisor: Ing. Jiří Bárta, Ph.D.

České Budějovice 2020

Kroll, L., 2020: Testing of different quality filtering algorithms on contig quality in plant transcriptome. Bc. Thesis, in English -26p., Faculty of Science, University of South Bohemia, České Budějovice, Czech Republic and Faculty of Engineering and Natural Sciences, Johannes Kepler University, Linz, Austria.

Abstract:

RNA sequences of the carnivorous plant Utricularia vulgaris were used to test the importance of RNA sequence quality control tools and the influence of those tools on a sequence assembly. Three quality control tools were examined, followed up by the testing of two different assemblers. The results were then compared and evaluated.

I hereby declare that I have worked on my bachelor thesis independently and used only the sources listed in the bibliography. I hereby declare that, in accordance with Article 47b of Act No. 111/1998 in the valid wording, I agree with the publication of my bachelor thesis, in full form resulting from deletion of indicated parts to be kept in the Faculty of Science archive, in electronic form in a publicly accessible part of the IS STAG database operated by the University of South Bohemia in České Budějovice accessible through its web pages.

Further, I agree to the electronic publication of the comments of my supervisor and thesis opponents and the record of the proceedings and results of the thesis defence in accordance with aforementioned Act No. 111/1998. I also agree to the comparison of the text of my thesis with the Theses.cz thesis database operated by the National Registry of University Theses and a plagiarism detection system.

In České Budějovice, 20.5.2020

[illegible]

....
Signature

Acknowledgements

I would like to take this opportunity to thank my supervisor Ing. Jiří Bárta, Ph.D. who enabled me to work on this interesting project and provided guidance and expertise throughout the process. I would also like to express my gratitude to Jiří Pech Ph.D. whose great work and assistance was very useful in cases of technical problems.

Table of Contents

1. Introduction

Transcriptome assemblies are a useful tool in researching organisms. A variety of analyses are based on transcriptome assemblies, like “differential gene expression analysis, metabolic pathway reconstruction, novel gene discovery or metabolic flux analysis” (Voshall & Moriyama, 2018). With the new sequencing and assembly techniques it is possible to assembly a full transcriptome of an organism relatively fast (Voshall & Moriyama, 2018). A complete transcriptome assembly can give insights on a variety of biological and evolutionary questions (Hölzer & Marz, 2019).

In this project RNA sequences from samples from different tissues of the carnivorous plant *Utricularia vulgaris* were analysed. The plant *Utricularia vulgaris* is an aquatic carnivorous plant that catches its prey with ultra-fast suction traps (Vincent & Marmottant, 2011; Westermeier et al., 2017). It has a very small genome size, with very few studies focusing on the actively transcribed genes during plant growth. For performing any kind of analysis with RNA sequencing data it is important to first evaluate the quality of the sequence reads and perform certain procedures to improve the quality of the data before further sequence processing (e.g. assembly). In several steps before the actual analysis, during the collection of the data, several errors can be incorporated into the analysis. Especially the application of Next Generation sequencing (NGS) technologies, which were used for the sequencing of this dataset, leads to these errors (Patel & Jain, 2012). There is a number of sequence artefacts that are likely to be incorporated, like read errors, poor quality reads, and primer or adapter contamination (Patel & Jain, 2012). All these examples decrease the quality of the reads which can later lead to problems in the analysis. Especially for the later usage of this dataset it is very important to have good quality reads, because bad quality can negatively influence the results of for example sequence analysis and gene expression studies (Patel & Jain, 2012). Therefore it is very important to perform a good quality control, filtering and trimming of the sequence, in order to avoid time-intensive problems due to sequencing errors, in the later analysis. To show the importance of this quality control tools several of these tools are tested in this work on a small part of the previously mentioned dataset. The quality control is applied on the sequences and then the results of a sequence assembly of those corrected sequences is compared to the sequence assembly results of the same uncorrected sequences. Additionally to testing different quality control tools, several different assemblers with different algorithms and principles should be used to choose the best assembly pipeline and diminish the risk of possible faulty assemblies. Regarding the huge amount of algorithms and pipelines existing, this thesis aims to distinguish important steps in common quality control pipelines and evaluate different tools based on their differing approaches.

2. Literary Review

2.1. Comparison of different quality control pipelines

As mentioned before there are a variety of issues in regard of the quality of RNA sequences. To overcome these difficulties it is recommended to evaluate the quality of a RNA sequencing (RNA-seq) experiment which can be done on two independent levels (Li, Nair, Wang & Wang, 2015). The first technique is based on the raw sequencing reads and does not involve further processing, like sequence alignment (Li, Nair, Wang & Wang, 2015). Therefore it is often described as a “low-level” strategy, as it is assessing the condition of the data based on the efficiency of the sequencing technologies and might oversee the biological reliability of the data(Li, Nair, Wang & Wang, 2015). It is suggested that this part of the quality control alone can cause flawed understanding of the data (Sheng et al., 2017). Hence additional evaluation on “higher level” is recommended to ensure correct results, like additional statistic evaluation, comparison with similar datasets etc. (Li, Nair, Wang & Wang, 2015).
A common technique to evaluate the quality of raw sequences is to estimate the probability of an error for each specific base-call in the sequencing read (Brent & Green, 1998). Therefore Phred quality score (q) is used which is working with “log-transformed error probabilities” and is defined as following

$$q = -10 \times \log_{10}(p)$$

with p being the “estimated error probability for that base call” (Brent & Green, 1998). A highere quality score shows a higher probability of a correctly identified base at that position. In general, a q score above 30 stands for good quality, whereas below 20 suggests poor quality, with everything in between being acceptable (Li, Nair, Wang & Wang, 2015). After the calculation of the per nucleotide quality score it is common to compute the "average quality score per read" (Li, Nair, Wang & Wang, 2015). In fact there are several ways to use Phred score for quality filtering in RNA sequences. It is possible to use the average quality score per read and filter out reads under a specific threshold. Another possibility is to use a so called "sliding-window" technique, where a window of a specific size is selected and then moved over the read (Bolger, Lohse & Usadel, 2014). If the average quality score in this window drops under a certain value the region is filtered out of the sequence. Finally, the last option is to asses the "per tile sequence quality" as the sequencing is performed on flow cells which are separated into tiles. By assessing this feature it is possible to cut out bad regions of a tile if their quality scores are below a selected limit. Finally, the performance of these programs will be evaluated based on the effects of them on the assemblies as well as on the alignment, to make sure that not only the raw sequences are tested on condition but their biological reliability will be inspected as well.

2.2.Comparison of different assembly strategies

Once the quality of the sequences is assured, sequence assembly is the next step the data is going through. The process joins overlapping sequences together to form larger contigs (Saraswathy & Ramalingam, 2011). There are two main methods to create an assembly, one option is the *DeNovo* Asssembly, another is the Reference Assembly (Voshall & Moriyama, 2018). For the latter approach a reference genome/transcriptome is needed, which is not always available. In this paper, only *DeNovo* Assembly will be describe. In this assembly the contigs are based directly on the data retrieved from sequencing and quality control, without making use of reference sequences (Voshall & Moriyama, 2018). In the majority of *DeNovo* assembly programs De Bruijn graphs are used to create the assembly (Martin & Wang, 2011). First the assembly program creates k-mers from the reads which are later used to construct the De Bruijn graph (Tjaden, 2015). K-mers are short subsequences from the original reads of a given length k (Voshall & Moriyama, 2018). By overlapping these sequnces, with the help of the De Bruijn graph, the original sequences can be recreated (Voshall & Moriyama, 2018). With assemblies one major limitation has to be noticed, because of the necessity of a k-mer “to start at every position along the original sequence in order for the graph to cover the full sequence.” (Voshall & Moriyama, 2018). This problem may result in either of the two most common and detrimental assembly errors. When k-mers are shorter likelihood that the full sequence is covered rises but so does the likelihood that one k-mer “corresponds to multiple reads from multiple transcripts.” (Voshall & Moriyama, 2018). This issue can be avoided by choosing longer k-mers but at the same time the “entire sequence of some transcripts” may not be covered (Voshall & Moriyama, 2018). The first problem results in an error called “Chimeras”, the second in an error called “Fragmentation” (Voshall & Moriyama, 2018).

In general it is important to pick a good algorithm that will be able to create a high quality assembly, while not requiring too many computational resources.

The first aim of this project is to test programs for all of the three previously mentioned quality filtering approaches to use Phred score for quality filtering in RNA sequences.

The second aim is to test the influence of several assembly algorithms and pipelines and their influence on assembly results.

3. Methodology

First, the control tool FastQC was used, to evaluate the quality of the Illumina reads (Version 0.11.8,Andrews, 2010). FastQC is able to detect problems resulting from the sequencing as well as from the starting library material (Version 0.11.8,Andrews, 2010). As output FastQC creates a report of the tested sequences. In the report the all operations that were run are displayed, including with a summary and a short evaluation of the results (Version 0.11.8,Andrews, 2010). The report shows Basic Statistics of the processed fastq file, the Per Base sequence quality, Per Tile sequence quality, Per tile sequence quality, Per sequence quality scores and several more parameters. As the collected data of the experiments contains forward and reversed reads resulting from pair-end sequencing, there are two FastQC reports per read pair, which were used for a first evaluation of the quality of the data. Especially interesting in the FastQC reports were the Per tile sequence quality, as one of the quality filtering tools later is based on this plot. Errors commonly displayed here are “bubbles going through the flow cell” , “smudges on the flow cell” or “debris inside the flow cell” (Version 0.11.8,Andrews, 2010). Aim of the tool Filter by Tile, which will be described later, was especially to remove such errors as displaced here.

Afterwards, Fast Length Adjustment of Short reads (FLASH) was used to merge the paired end reads (Version 1.2.11, Magoč & Salzberg, 2011; The Center for Computational Biology at John Hopkins University [CCB], n.d.). By the use of this software longer reads are created which may improve the result of transcriptomic assemblies (CCB, n.d.). After Flash, FastQC was run again, but only on the samples that seemed particularly interesting for this project, in order to have a more detailed evaluation for the merged-reads. Instead of two FastQC reports per file, this number was diminished to only one FastQC report for the files, which were going to be analysed further.

As the first evaluation of the data, as well as the merging of forward and reversed reads has been created, the next step consists of the testing of several different quality control tools. Out of the high diversity of quality control tools for RNA sequences three different programs were created, with three different approaches to filter the data. For all three quality control tools, the output from Flash was used as the provided input sequences.

As a first quality control tool PRINSEQ was used for reformatting, filtering and trimming of the RNA sequences (Version 0.20.4, Schmieder & Edwards, 2011). As and addition, a short summary, which is created after the execution of the program, is provided (Version 0.20.4, Schmieder & Edwards, 2011). The tool is able to evaluate data on several parameters like, GC content, ambiguous bases, Poly-A/T tails or Taq sequences and filters, reformats and trims the sequences dependent on the selected options (Version 0.20.4, Schmieder & Edwards, 2011). In this case the tool was used to filter out sequences with a low average quality score. As PRINSEQ provides another summary about the filtered sequences, the final selection of files for the rest of the work was made. In total five different files were completely analysed, selected based on their FastQc reports as well as the results provided by PRIN-SEQ.

Alternatively to PRINSEQ the tool Trimmomatic was used on the five different files that were previously selected (Version 0.39, Bolger, Lohse & Usadel, 2014). For this tool the sliding window approach was used for filtering of the data, which scans reads from 5' to 3' direction and deletes the 3' end of the read if the average quality score of a specified group of bases, the window, falls under the described threshold (Version 0.39, Bolger, Lohse & Usadel, 2014).

As a third option for quality filtering, Filter By Tile was used, which is part of the package BBMap (Version 38.21, Bushnell, 2014). The main part of this software consists of an alignment tool, but the incorporated script Filter By Tile, is a quality control tool that uses the per tile sequence quality score for filter of sequencing reads.

Finally, to evaluate the success of the quality control and filtering, assemblies were created using two different assemblers. For all five reads, the output sequences of first Flash, then Prinseq, Trimmomatic and finally FilterByTille were run.

The first assembler, that was run for evaluation was Velvet, an assembly program using de Brujin graphs mainly used for genetic sequence assembly "ideal for high coverage, very short read (25-50bp) data sets" (Version 1.2.10, Zerbino & Birney, 2008). Velvet especially was created with the aim to eliminate experimental errors and repeats, by using a two step approach (Version 1.2.10, Zerbino & Birney, 2008). First, sequencing merging performed by an "error correction algorithm", then a "repear solver" to distinguish "paths sharing local overlaps (Version 1.2.10, Zerbino & Birney, 2008).

To also test a second approach for the assembly creation the software MEGAHIT was used (Version1.2.9, Li et al., 2015). This software utilises "succinct de Bruijn graphs which are a compressed version of de Bruijn graphs" (Version1.2.9, Li et al., 2015). MEGAHIT especially was created with the aim to "assemble large and complex" dataset, while efficiently managing good performance with as little as possible expensive resources (Version1.2.9, Li et al., 2015). The tool focuses on created high quality assemblies in a short time, with relatively small computer memory usage (Version1.2.9, Li et al., 2015).

Following the successful completion of assemblies for all files and all different quality control and filtering tools, the results from both Velvet and MEGAHIT were evaluated using the Quality Assessment Tool QUAST (Version 5.0.2, Gurevich et al., 2013). QUAST computes various metrics for evaluation of assemblies and visualises them for easier comparison of different assemblies (Version 5.0.2, Gurevich et al., 2013).
After all the QUAST evaluations for the assemblies were created, the output was used to create further statistics to compare the results and the differences resulting out of different quality control and filtering as well as varying assembly strategies.

4. Results

4.1. Selection of Files

As one of the first steps of the project five files had to be selected to be tested in full. In total the complete dataset included 18 different files from different tissues of the plant. Aim of the selection was to show the different behaviour of the tools dependent on the differing files. Some of the earlier mentioned tests were run beforehand on the full dataset, to help with the final selection. In order to easily identify them in further research the files were not renamed in this project. To display the highest amount of diversity, the chosen files have differing length and based on first evaluation a very differing quality based on different criteria. This can be observed in the following sections.

4.2. FastQC and Flash

FastQC creates a report for every tested file that can be viewed in html format and downloaded. In the beginning of the report a short summary with basic statistics about the tested dataset is shown, containing the number of total sequences, the amount of sequences flagged as poor quality, the sequence length and finally the GC content as a percentage. A short summary, is shown in the figure below (Tab. 1). There only forward reads (R1) are considered as the basic summary is very similar for the forward and reversed reads (R2) , with only the GC content slightly differing between R1 and R2 for each file. From this part of the analysis the total amount of sequences was most important, as it was crucial to select differing files, in order to be able to evaluate how the file size might impact later results.

	File 6- R1	**File 8- R1**	**File 10- R1**	**File 11- R1**	**File 15- R1**
Total sequences	17720809	54193881	6317255	22152877	35392415
Sequences flagged as poor quality	0	0	0	0	0
Sequence length	101	101	101	101	101
% GC	44	44	42	46	47

Table 1: Basic statistics of all five tested fasqt files - forward reads (R1)

The next part of the report illustrated the per base sequence quality. Quality scores highlighted green in the graph illustrate good scores across these position, whereas quality scores printed in red illustrate bad scores across these positions. Above the graph there is also a small icon indicating wether the per base sequence quality appears to be acceptable or not. For all files, forward as well as reversed reads, the data in the graphs was mainly illustrated in the green part, indicating a good per base sequence quality for all files. As an example, the graph for the File 6 - R1 is added in thee figure below (Fig 1).

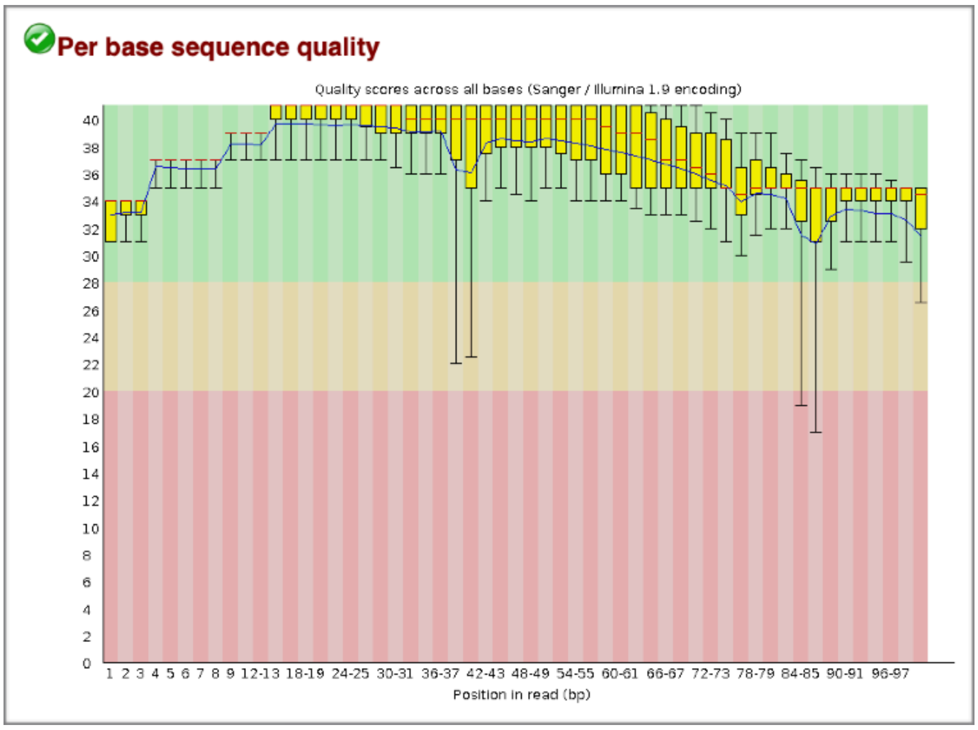

Fig 1: Per base sequence quality for File 6- R1

As the data for this project were sequenced by an Illumina sequencing machine the next part of the report shows the per tile sequence quality. The plot displays the “deviation from the average quality for each tile” (Version 0.11.8,Andrews, 2010). In case a tile shows a “Phred score more than 2 less than the mean for that base across all tiles” a warning will be displayed (Version 0.11.8,Andrews, 2010). Overall majority of the plot should be blue when the tiles have good quality, with warmer colours displaying worse quality. In this dataset the reports of all files, with both forward and reversed reads, issued a warning in this part of the report. With some files showing worse results than others. As displayed the merging of sequencing with FLASH in the next step improved the results for some files a little, but the overall per tile sequence quality is still problematic. This is shown here with the example of dataset 15, were R1(Fig 2A) and R2 (Fig 2B) both display problematic regions in the plot. The merging of these two reads displays improvement in quality but as visible in the last figure (Fig 2C), the problematic regions are still noticeable.

The per sequence quality scores show a good distribution for all files and reads and don’t show any sign of errors, like air-bubbles.

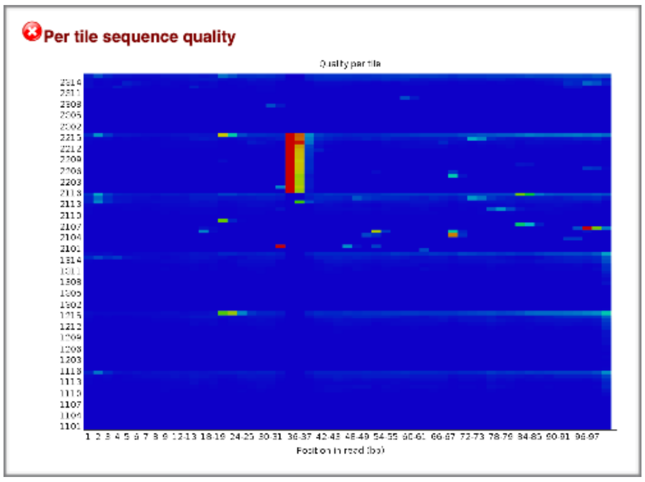

Fig 2A: Per tile sequence quality for file 15 - R1

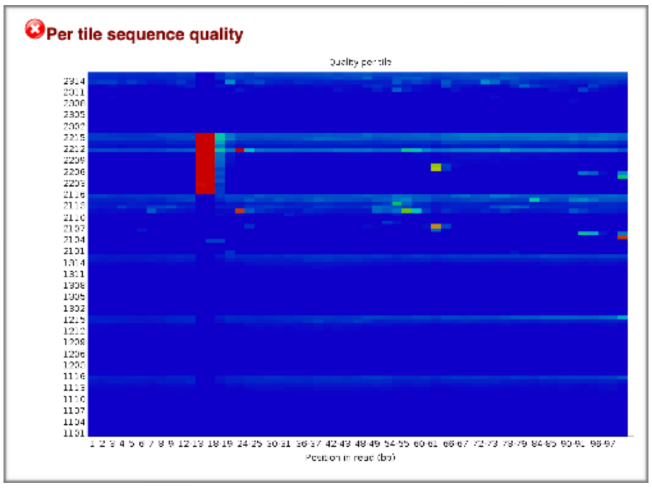

Fig 2B: Per tile sequence quality for file 15 - R2

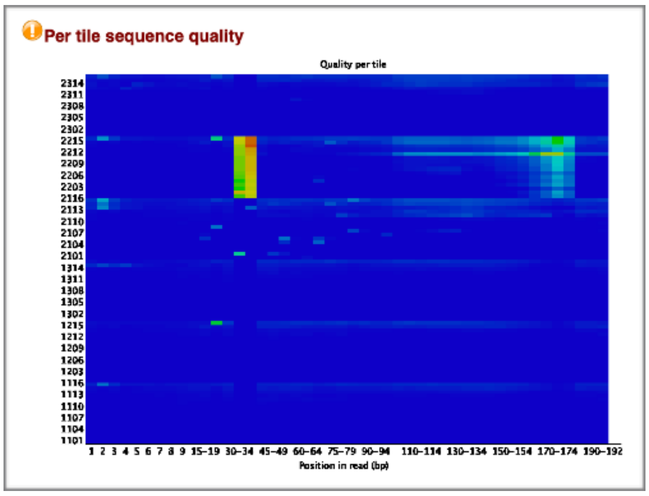

Fig 2C: Per tile sequence quality for file 15 - with reads merged via Flash

In contrast to this the per base sequence content showed several warnings. However, none of these warning were red. In this part, most files had relatively similar results, showing only a little variation. As an example the plot for file 10- R1 is provided in the figure below (Fig 3).

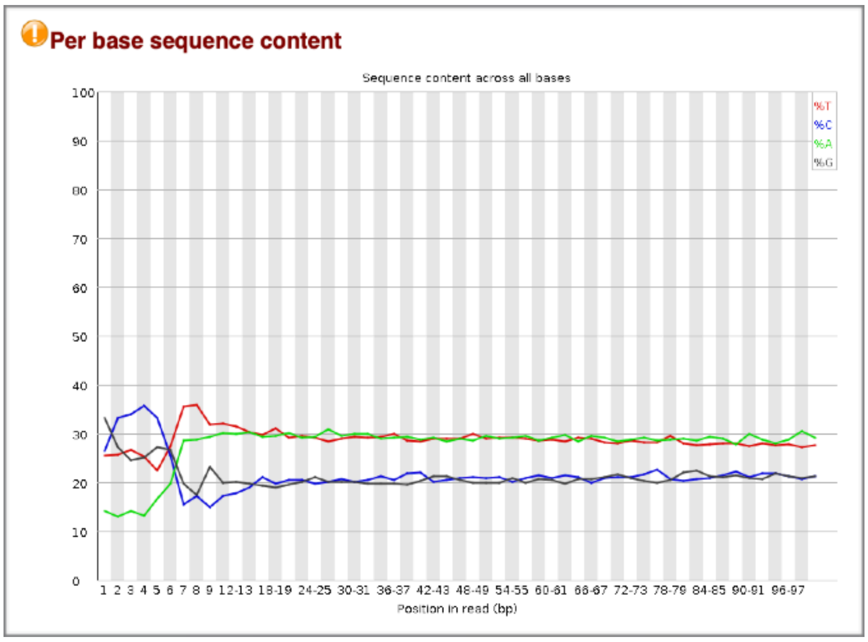

Fig 3: Per base sequence content for file 10 - R1

The per sequence GC content results show significant variation across all files. For all of them a warning was issued, in either yellow or red. Even after Flash the results did not improve for the files, as both reads showed more or less the same distribution. The two figures below illustrate the differences between two files, and how diverse these are for different parts of the dataset. In the first figure, the GC distribution of file 8 - R1 is displayed, where the distribution is still quite similar to the theoretical one (Fig 4A). The graph of file 15 - R1 on the other hand, shows a distribution differing quite significantly to the theoretical distribution (Fig 4B).

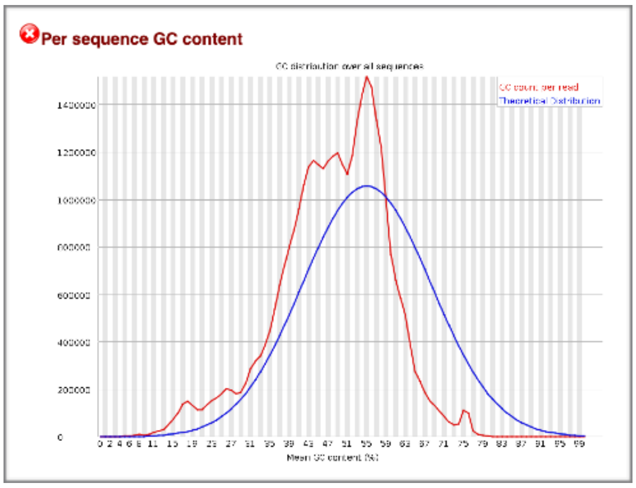

Fig 4A: Per sequence GC content for file 8 - R1

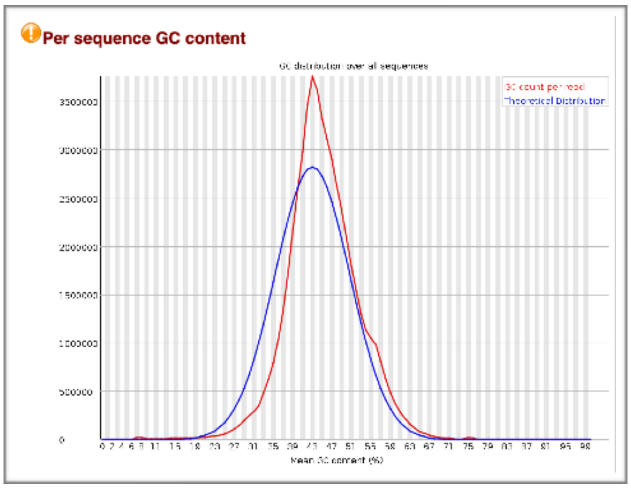

Fig 4B: Per sequence GC content for file 15 - R1

The next point in the report, the per base N content, appears to be more or less unobtrusive for all five files, with only small signs of variation across the datasets.
After this, the Sequence length distribution, is again similar for all five files. In none of the reports a warning was issued, indicating good quality sequences in regard of these values.
Another part of the report, which illustrated higher variation across all files is the Sequence Duplication level. Most files except for file 10 (Fig 5A) showed a red warning here, as visible in the figures below on the example of file 8(Fig 5B).

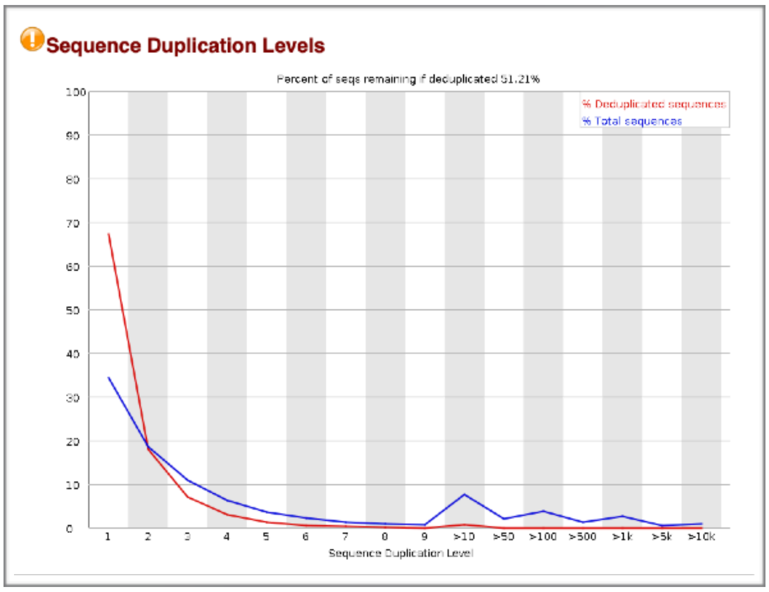

Fig 5A: Sequence Duplication levels for file 10 - R1

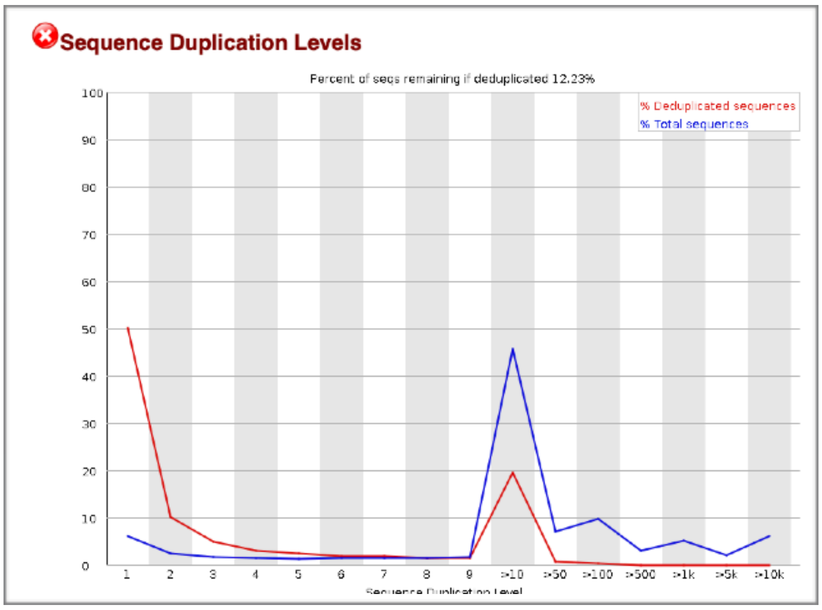

Fig 5B: Sequence Duplication levels for file 8 - R1

The FastQC reports also contained lists with overrepresented sequences, displayed with a count, a percentage and a possible source. Majority of the files issued a warning here. However the only file that displayed a red warning was file 6 - R1.

The last part of the report, illustrated the adapter content for several specific adapters. In this case, the presence of none of these adapters could be recognised.

In general it is visible that all five files, display significant difference in quality measured on the previously mentioned assessed parameters. In the following part of the work other quality control and filtering methods were applied, in order to eliminate some of the errors illustrated by the FastQC reports.

4.3. Results from the quality control tools

4.3.1. Prinseq

The first quality filtering tool that was tested was Prinseq. It was run with the usage of the automatically applied settings. As output, Prinseq delivers the trimmed and filtered sequences in fastqc format. Additionally a summary is provided to illustrate the amount of sequences that were filtered. In the table below the main parts of this summary for all five files is provided (Tab. 2). According to Prinseq File 8 has the smallest amount of bad sequences whereas the File 6 had the largest amount of bad sequences. The other files, 10, 11 and 15 all had a similar amount of bad sequences, around 20 percent.

	File 6	File 8	File 10	File 11	File 15
Input sequences	2 490 593	18 524 857	3 133 942	4 642 630	13 026 613
Bad sequences	730 506	2 471 606	725 256	1 031 097	2 371 940
Bad sequences in %	29.33	13.34	23.14	22.21	18.21

Table 2: Summary of the Prinseq output for all five fastq files

4.3.2. Trimmomatic

The tool Trimmomatic was used to test the sliding window approach of quality filtering tools. The size of the window was set to 50 base pairs (bp) and the required quality score to 25. In general, it seems like Trimmomatic only filtered out a very small amount of reads for all files, by application of the previously specified parameters, as the percentage of dropped reads always stayed significantly under one. A summary of the amount of filtered out reads is provided in the figure below (Tab. 3).

	File 6	File 8	File 10	File 11	File 15
Input reads	2 490 593	18 524 857	3 133 942	4 642 630	13 026 613
Dropped reads	3 612	26 089	5 447	7 653	54 598
Dropped reads in %	0.15	0.14	0.17	0.16	0.42

Table 3: Summary of the Trimmomatic output for all five fastq files

4.3.3. Filter by Tile

As a last quality filtering tool Filter by Tile was applied. Because it is based on the per tile sequence quality, FastQC was used to visualise the differences between the original Flash file, and the file after filtering it with this software. As visible in the figure below, the Per Tile sequence quality graph shows more red regions as it did before. It suggests that particularly those parts of the file have been fully deleted due to bad per tile sequence quality. The first figure (Fig 6A) shows the FastQC per tile sequence quality for file 6, with merged reads R1 and R2, whereas the second figure (Fig 6B) shows the per tile sequence quality of the same file after application of Filter by Tile. It can be seen how especially in the region where the per tile sequence quality of file 6 displayed problematic regions, a large amount of reads were deleted.

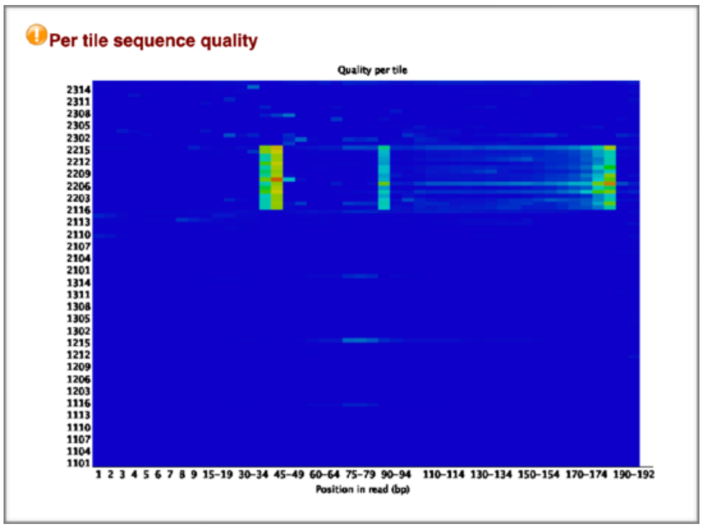

Fig 6A: Per tile sequence quality of file 6 with merged reads

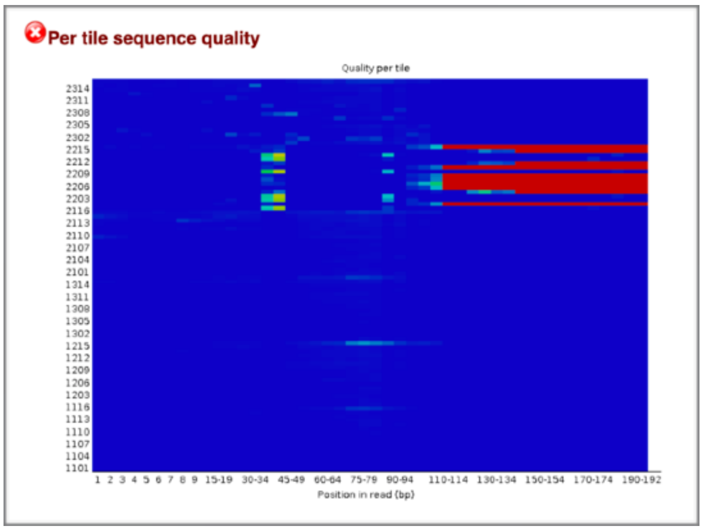

Fig 6B: Per tile sequence quality of file 6 after application of Filter

4.4.Comparison of the different assemblies for all quality filtering tools

After the quality control all resulting sequencing files were assembled using Velvet and MEGAHIT. Both assemblers were run with the preset parameters from the software. The resulting fasta files were then evaluated using QUAST. The output of this evaluation tool is described in the following part of the thesis, ordered by file. QUAST created statistics that include the number of contigs, their length and several other parameters. In the following three values were used to illustrate differences between quality filtering tools and assembly algorithms. First, the total number of contigs was observed, regardless of their length. Second, the amounts of contigs with a size bigger or equal to 500 bp were compared. Finally, the lengths of the largest contigs created by the assemblies was evaluated. For better visualisation, several graphs were constructed.

4.4.1. Differences in contig amounts

For File 6 the first significant difference that is observable is the number of contigs created by Velvet compared to MEGAHIT. Velvet created a significantly larger amount of contigs than Megahit (Fig 7A). This discrepancy changes with the size of contigs increasing. For contigs larger or equal to 500 bp MEGAHIT produces notably more contigs than Velvet (Fig 10B). This is one of the differences, observable for all five files. Velvet constructs a very large amount of contigs, which are however relatively small. Even though MEGAHIT overall assembles less contigs, the majority of them is larger. The largest contig is also in all five files created by MEGAHIT, as is also visible on the example of file 6 (Fig 7C).

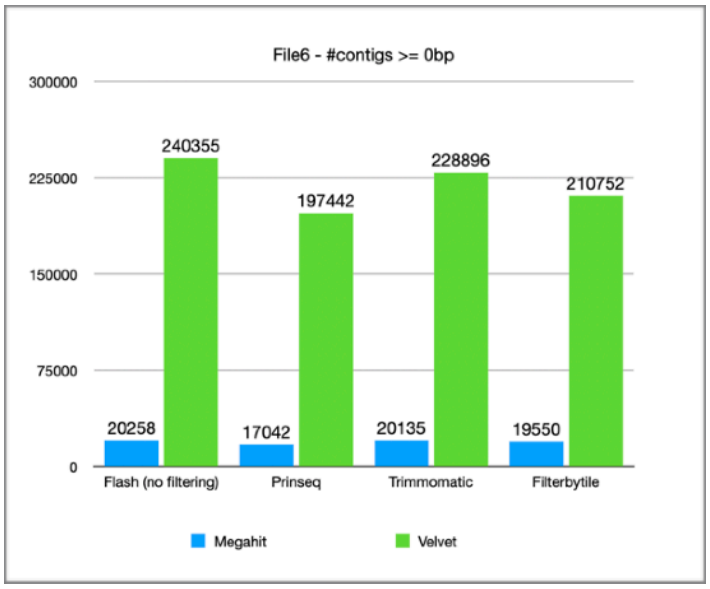

Fig 7A: Total amount of contigs with a size equal or larger to 0 bp for file 6

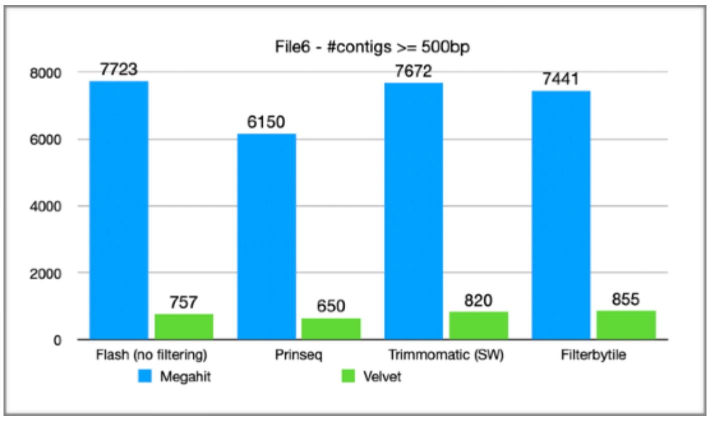

Fig 7B: Total amount of contigs with a size equal or larger to 500 bp for file 6

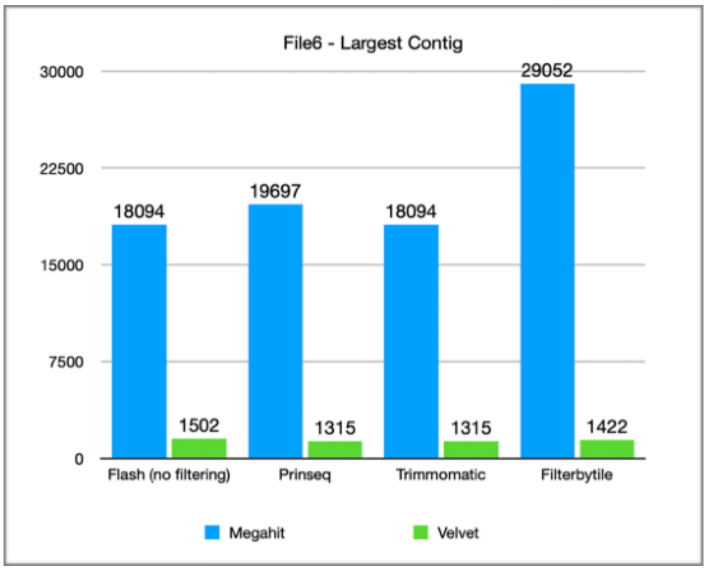

Fig 7C: Largest contigs for file 6

Moving further to the quality control tools, it becomes clear that the amount of contigs diminished by application of any of the previously mentioned quality control tools. This observation holds for the total amount of contigs (Fig 7A), as well as for the amount of contigs equal or larger than 500 bp (Fig 7B). The most noticeable difference between quality control tools can be seen with the change in size of the largest contig assembled by MEGAHIT. After application of Filter By Tile, the largest contig grew significantly larger as can be observed in the figure below (Fig 7C). This improvement however, can not be severed for the assembly produced by Velvet.

4.4.2. Increase in largest contig sizes

For File 8 the observations made, were quite similar to the ones from File 6. Velvet creates the larger amount of contigs in general (Fig 8A), whereas MEGAHIT produces larger contigs (Fig 8B) as well as a significantly higher value of the largest contig assembled. Here it is especially noticeable how the quality filtering produced by PRINSEQ always results in a decrease in values, even in comparison to the completely unfiltered version. Again, especially for the value of the largest contig assembled by MEGAHT a significant increase after application of Filter By Tile can be observed (Fig 8C). Whereas PRINSEQ as well as Filter by Tile, seem to change the result of the assemblies so far, the same can not be said about Trimmomatic. For File 6 as well as for File 8 the application of Trimmomatic does not change the results very much, they are more or less equal or at least comparable to the assemblies from the files without filtering.

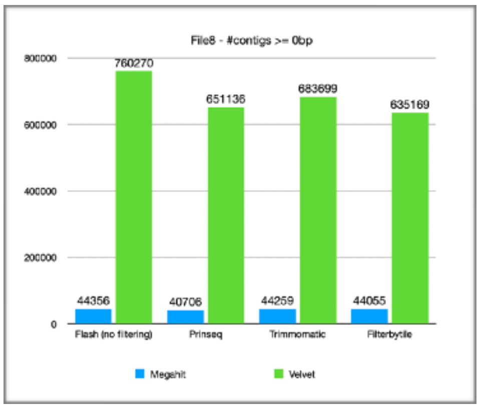

Fig 8A: Total amount of contigs with a size equal or larger to 0 bp for file 8

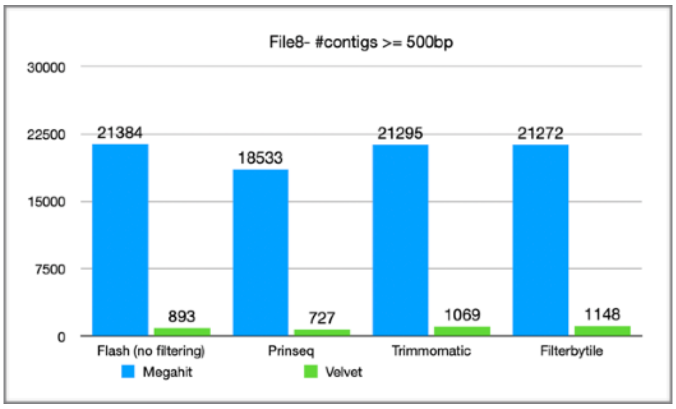

Fig 8B: Total amount of contigs with a size equal or larger to 500 bp for file 8

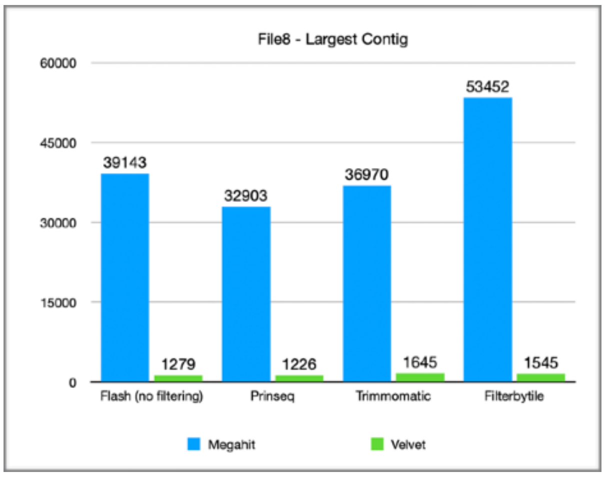

Fig 8C: Largest contigs for file 8

4.4.3.Differences between different samples

For File 10 most of the observations previously made still apply. The only noticeable difference is that the growth of the largest contig assembled by MEGAHIT after application of Filter by Tile can not be observed. Instead the values stays similar as the ones no application of a quality control tool or after the application of either PRINSEQ or Trimmomatic (Fig 9).

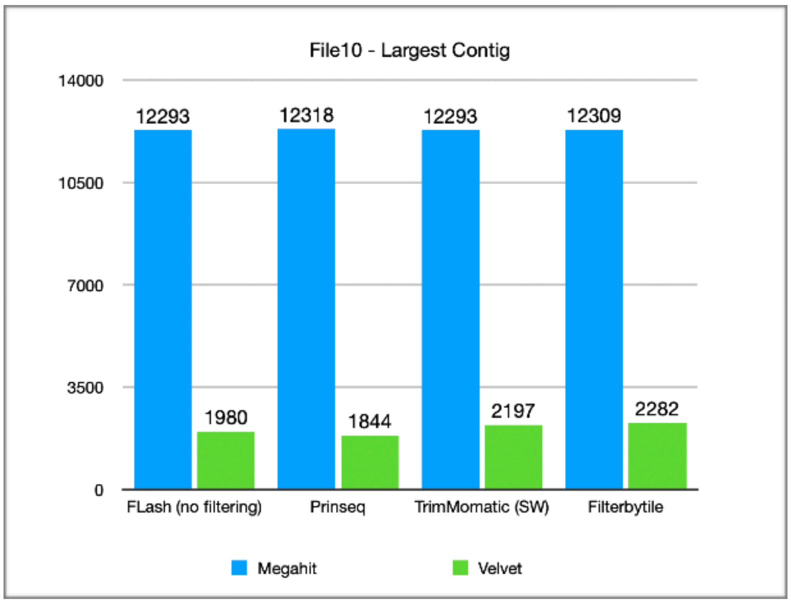

Fig 9: Largest contigs for file 10

4.4.4. Comparison to previous results

The results from file 11, seem very similar to the previous ones, which is the reason for no provided figure in this case. Again, similar to file 10, a growth of the largest contig assembled by MEGAHIT after application of Filter by Tile can not be observed.

4.4.5.Number of contigs differences and comparison of N50

Finally the results for File 15 are again, very similar to the two files, File 10 and File 11, from before. The only difference here is visible in case of the total number of contigs. Here the number of contigs assembled by Velvet is significantly smaller when no quality filtering is applied (Fig 10). This however is only the case for contigs bigger or equal to a size of 0 bp not for contigs bigger or equal the size of 500 bp.

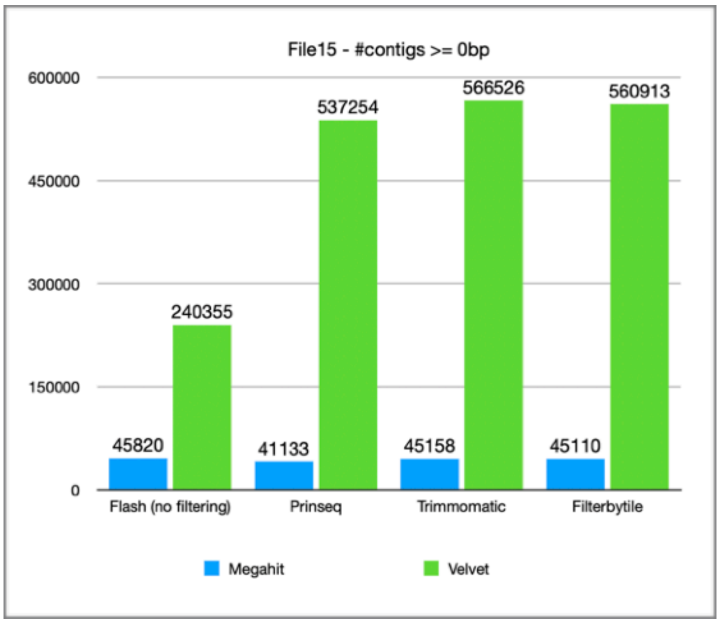

Fig 10: Largest contigs for file 15

To get even better inside in the different results from the assemblies. The N50 size, which is also included in the QUAST evaluation, was used for further comparison. The N50 “equals the length of the scaffold (or contig) overlapping the midpoint of the length-order concatenation of scaffolds (contigs).” (Mäkinen, Salmela & Ylinen, 2013).As visible in the figure below for file 15 (Tab. 4) the N50 values for MEGAHIT are significantly larger than the ones for Velvet. Again, the application of the quality control and filtering tools, did not change the N50 values much. For Velvet there was a small value increase after quality filleting, for Megahit in the case of application of PRINSEQ the value even grew smaller. In regard of the other files, a similar observation can be made while evaluating the N50 values. MEGAHIT usually has a notably higher one compared to the value produced by Velvet. The application of quality filtering on the other hand, creates only small variation in the data.

	Flash (no filtering)	Prinseq	Trimmomatic	Filter by Tile
MEGAHIT	820	800	822	819
Velvet	586	599	602	607

Table 4: N50 statistics for file 15

5. Discussion

First of all, all results show quite significant differences in their assemblies provided by either MEGAHIT or VELVET. Where VELVET produces a larger amount of small contigs, MEGAHIT creates viewer but longer ones. This output may be due to some basic limitations from the software. VELVET is a program created to assemble very short reads, with a relatively short k-mer size (Version 1.2.10, Zerbino & Birney, 2008). On the other hand, it should be able to create quite long assemblies as well (Version 1.2.10, Zerbino & Birney, 2008).
Another noticeable difference between the two tested assemblers is the discrepancy in their N50 values. N50 is “a typical measure to assess how well the assembly has succeeded” (Mäkinen, Salmela & Ylinen, 2013). Across all files, Velvet produces significantly smaller N50 values than MEGAHIT.
In regard to usability, both MEGAHIT as well as Velvet are easy to work with, with clear instructions provided in the manual. Most of the time MEGAHIT performed the assembly faster than Velvet, this however did not seem significant in this case. The faster performance might only be considered, in case of a very large dataset, to safe resources.
In summarisation, the assemblies produced by Velvet seemed to be worse than those produced by MEGAHIT. At least on this dataset, MEGAHITs aim to produce larger and faster assemblies seemed to be confirmed (Version1.2.9, Li et al., 2015). Of course, it is necessary to note that MEGAHIT is a newer program and therefore might benefit from a more successful implementation of its algorithms.

In regard to the quality control and filtering, it is important to take the first realisation about the data in regard. Overall the sequences seemed to have a high overall quality, as can be easily observed in the exceptionalness good outputs of the per base sequence quality in the FastQC reports. This may have resulted in very little filtering performed from especially Trimmomatic and PRINSEQ. As described earlier, Trimmomatic filtered out under one percent of reads for each file. This means that the average quality in a window of 50 bp was above the threshold in majority of cases. Therefore, this indicates a very good quality dataset again. On the other hand, it is also explanation why, the results of the assemblies were most of the time quite similar to those created after no application of any quality control and filtering tool. Finally, for file 15 were Trimmomatic performed the largest amount of filtering in its performance over the dataset, it is interesting to note, how different the results look from the unfiltered version in this case. This result may suggest a significant improvement of quality for file 15. In total, it is possible that Trimmomatic, did not apply a lot of filtering for most files as the dataset by itself had very good quality in most cases.

Even though PRINSEQ filtered out more data than Trimmomatic, the results are still very similar to the unfiltered ones, a lot of times even worse. The filtering applied by Prinseq may have been a little too strict resulting in not only faulty sequences but also good ones to be deleted.

As suggested in literature the most important part when it comes to quality control of RNA sequencing data is a good sample collection and well performed sequencing and not necessarily the quality control performed by software afterwards (Sheng et al., 2017). In this case the dataset seemed to have a good overall quality, and majority of the filtering applied later on was not inevitably necessary.

The only part where this dataset showed signs of inferior quality were the per tile sequence qualities. The data displaced by FastQC showed signs of errors, like “bubbles going through the flow cell” (Version 0.11.8,Andrews, 2010). With the application of the quality control tool Filter by Tile, a possible improvement of these errors was tested. The improvement of the size of the largest contig for both file 6 and file 8 suggest that such a correction has been made. As the results for other files are differing to this observation and the N50 values for both file 6 and file 8 do not show a similar improvement after application of Filter by Tile it is difficult to evaluate wether the program made a significant difference.

6. Conclusion

In general, it can be said that especially the selection of a suitable assembly program for the specific dataset is very important. There are vast differences in created assemblies by different tools, and as the assembly is a relatively time consuming step, the selection should be made with caution and research. At this point, majority of assemblers have advantages for some specific kind of dataset and disadvantages for another. Therefore, it is advisable to identify the dataset first, and collect information about it before selecting a suitable assembly program. A tool like FastQC can be very useful, to have a first look on the data to come to a decision how the particular sequences can be analysed best, and if quality control has to be applied. Additionally, it may help to select the right quality control tool, that will show most significant improvement for the dataset.
Further research is needed to determine effects of other assemblers on the results or to test other quality control tools. It would be also interesting to apply a similar pipeline on a dataset, that shows signs of notably worse quality as that may lead to more differences between filtered and unfiltered data.

7. References

Andrews, S. (2010). FastQC: A Quality Control Tool for High Throughput Sequence Data [Online]. Available online at: http://www.bioinformatics.babraham.ac.uk/projects/fastqc/

Bolger, A. M., Lohse, M., & Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina Sequence Data. Bioinformatics, btu170

Brent, E., & Green , P. (1998). Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities. Genome Research , *8*(3), 186–194. doi: 10.1101/gr.8.3.186

Bushnell, B..(2014). BBMap: A Fast, Accurate, Splice-Aware Aligner. United States.

Gurevich, A., Saveliev, V., Vyahhi, N., & Tesler, G. (2013). QUAST: quality assessment tool for genome assemblies. Bioinformatics (Oxford, England), 29(8), 1072–1075. https://doi.org/10.1093/bioinformatics/btt086

Hölzer, M., Marz, M. (2019). *De novo* transcriptome assembly: A comprehensive cross-species comparison of short-read RNA-Seq assemblers, GigaScience, 8(5), giz039, https://doi.org/10.1093/gigascience/giz039

Li D, Liu CM, Luo R, Sadakane K, Lam TW. (2015). MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. Bioinformatics (Oxford, England). 31(10), 1674-1676. doi: 10.1093/bioinformatics/btv033.

Li X., Nair A., Wang S., Wang L. (2015). Quality Control of RNA-Seq Experiments. In: Picardi E. (eds) RNA Bioinformatics. Methods in Molecular Biology, 1269. Humana Press, New York, NY

Magoč, T., & Salzberg, S. L. (2011). FLASH: fast length adjustment of short reads to improve genome assemblies. Bioinformatics (Oxford, England), 27(21), 2957–2963. doi:10.1093/bioibtr507nformatics/

Martin, J., Wang, Z. (2011). Next-generation transcriptome assembly. Nat Rev Genet 12**,** 671–682. https://doi.org/10.1038/nrg3068

Mäkinen, V., Salmela, L. & Ylinen, J.(2013). Normalized N50 assembly metric using gap-restricted co-linear chaining. BMC Bioinformatics, 255 . https://doi.org/10.1186/1471-2105-13-255

Patel RK, Jain M (2012) NGS QC Toolkit: A Toolkit for Quality Control of Next Generation Sequencing Data. PLoS ONE 7(2): e30619. https://doi.org/10.1371/journal.pone.0030619

Saraswathy, N., Ramalingam, P. (2011).8 - Genome sequence assembly and annotation. In: Concepts and Techniques in Genomics and Proteomics.109-121. https://doi.org/10.1533/9781908818058.109.

Schmieder, R., & Edwards, R. (2011). Quality control and preprocessing of metagenomic datasets. Bioinformatics (Oxford, England), 27(6), 863–864. https://doi.org/10.1093/bioinformatics/btr026

Sheng, Q., Vickers, K., Zhao, S., Wang, J., Samuels, D. C., Koues, O., Shyr, Y., & Guo, Y. (2017). Multi-perspective quality control of Illumina RNA sequencing data analysis. *Briefings in functional genomics*, *16*(4), 194–204. https://doi.org/10.1093/bfgp/elw035

The Center for Computational Biology at Johns Hopkins University[CCB]. (n.d.). FLASh**.** Retrieved August 28, 2019 from https://ccb.jhu.edu/software/FLASH/

Tjaden, B. (2015). *De novo* assembly of bacterial transcriptomes from RNA-seq data. Genome Biol 16, 1. https://doi.org/10.1186/s13059-014-0572-2

Vincent Olivier & Marmottant Philippe (2011) : Carnivorous Utricularia: The buckling scenario, Plant Signaling & Behavior, 6:11, 1752-1754, doi: 10.4161/psb.6.11.17804

Voshall A. & Moriyama E. (2018). Next-Generation Transcriptome Assembly: Strategies and Performance Analysis, Bioinformatics in the Era of Post Genomics and Big Data, Ibrokhim Y. Abdurakhmonov, IntechOpen, doi: 10.5772/intechopen.73497.

Westermeier Anna Sofia, Fleischmann Andreas, Müller Kai, Schäferhoff Bastian, Rubach Carmen, Speck Thomas & Poppinga. (2017). Trap diversity and character evolution in carnivorous bladderworts (Utricularia, Lentibulariaceae), www.nature.com/scientificreports

Zerbino, D. R., & Birney, E. (2008). Velvet: algorithms for de novo short read assembly using de Bruijn graphs. Genome research, 18(5), 821–829. https://doi.org/10.1101/gr.074492.107